Project 2

1.UML class diagram

Tile

-num : Integer

-Tlabel : QLabel \*

<<constructor>>+Tile()

<<destructor>>+~Tile()

+getNumber() : Integer

+getLabel() : QLabel\*

+setNumber(number : Integer)

+setText()

+setNumTxt(number : Integer)

+setLabel(label : QLabel\*)

Qwidget

↑

game2048

-ui : game2048

-palette : QPalette

-game2048bg : QImage

-tilesMap : QVector< QVector<Tile\*> >

-tilesCord : QVector< QVector<QPoint\*> >

-tilesbg : QVector< QVector<QLabel\*> >

-score : Integer

-bestscore : Interger

-tempscore : Interger

-Twidth : Interger

-Theigth : Interger

-emptyPointsList : QList<QPoint>

-slideStartList : QList<QPoint>

-slideEndList : QList<QPoint>

-mergeList : QList<QPoint>

- gameoverlabel : QLabel\*

-youwin : QLabel\*

-tempscorelabel : QLabel\*

-gameoverpic : QLabel\*

-youwinpic : QLabel\*

-first : bool

-aniSMfinished : bool

-aniRfinished : bool

-ifwin : bool

<<constructor>>+game2048(parent : QWidget \* = 0)

<<destructor>>+~game2048()

+initgame()

+setupTiles()

+startGame()

+showBestScore()

+rndaddNum()

+slideLeft()

+slideRight()

+slideUp()

+slideDown()

+addTempScore(s : Integer)

+addScore()

+showScore()

+tempscoreAni()

+saveBestScore()

+slideAni()

+mergeAni()

+checkGameOver()

+frndaddNum()

+finishSlide()

+finishMerge()

-on\_PBNewGame\_clicked()

-showEvent( : QShowEvent\*)

-resizeEvent( : QResizeEvent\*)

-keyPressEvent(event : QKeyEvent\*)

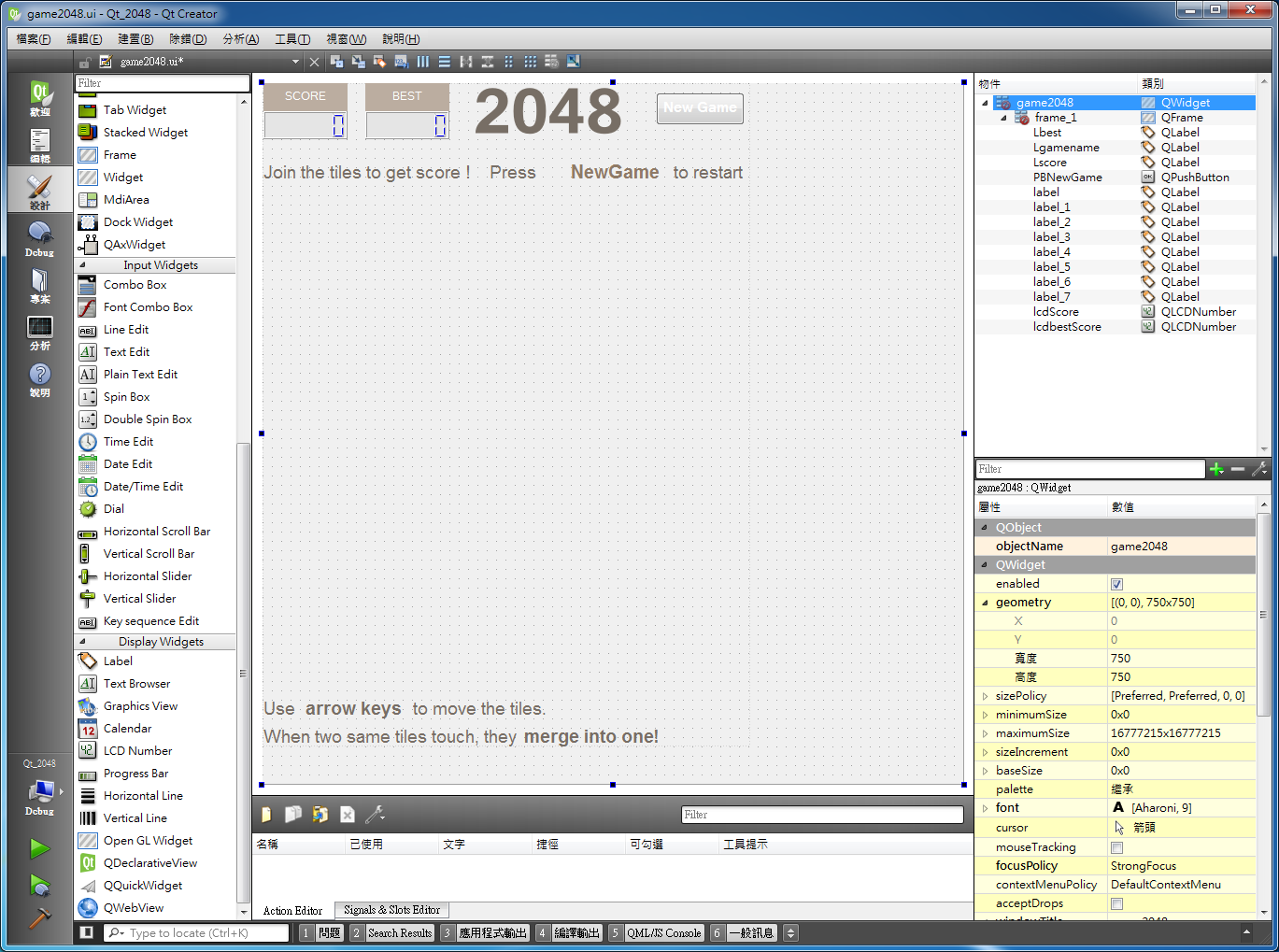
2.遊戲開發過程概述

類別名稱取名game2048﹒因為不需要Menu Bar﹑Tool Bars等﹐所以基礎類別選擇QWidget

接著用Qt Designer來設計和建立UI

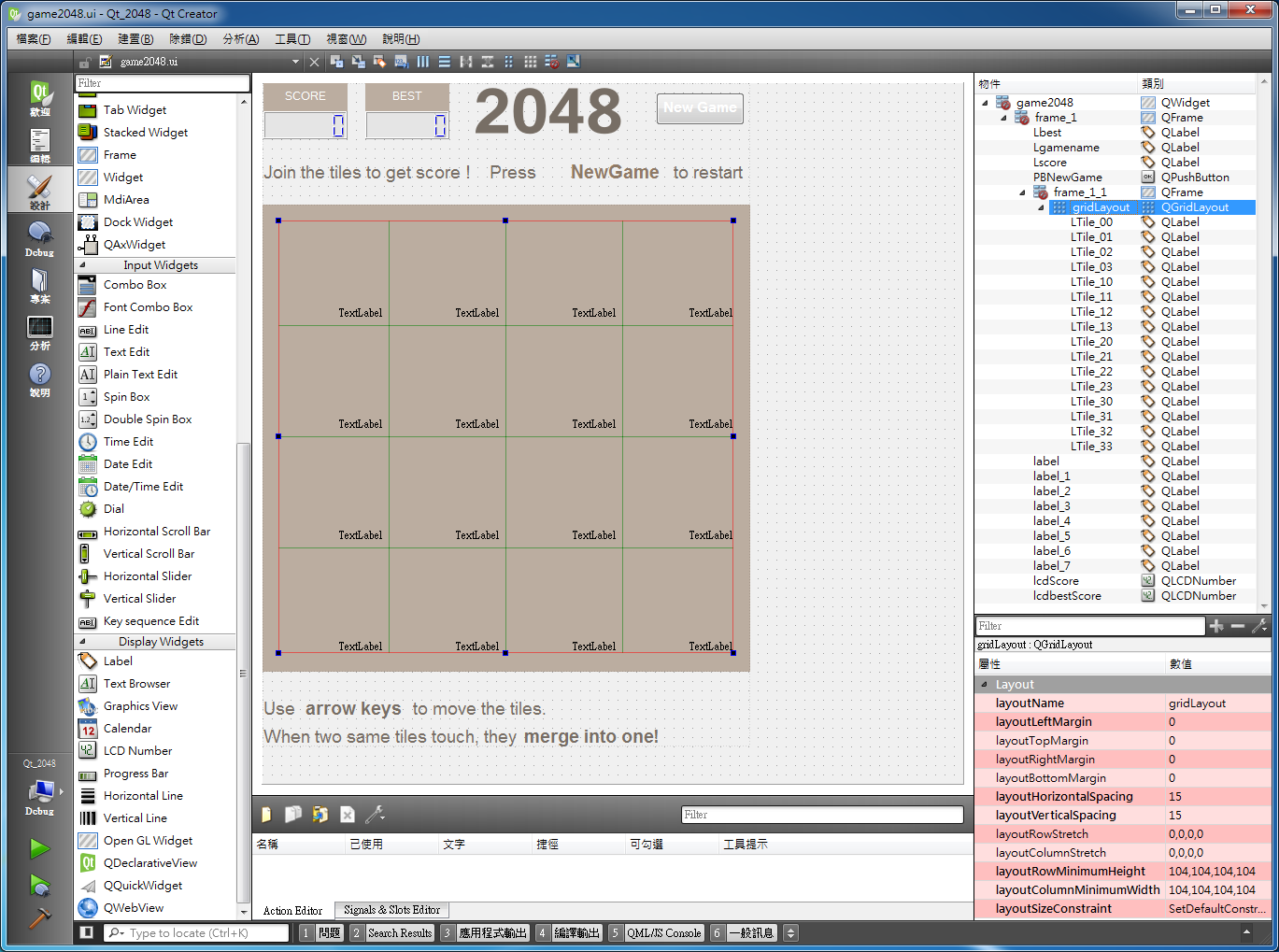
將game2048 QWidget視窗調整到適當大小﹒當玩家調整遊戲視窗尺寸時﹐程式需改變遊戲裡各物件的位置﹒所以先放一個Frame﹐取名Frame\_1﹐再把所有物件放在Frame\_1上﹒這樣只要改變Frame\_1的位置﹐在Frame\_1裡的所有物件就改變位置了﹒接著在Frame\_1左上角佈署1個顯示固定內容“SCORE”的Label﹐取名Lscore﹒在Lscore下面佈署1個顯示遊戲分數的LCD Number﹐取名lcdScore﹒然後在Lscore 的右邊佈署1個顯示固定內容“BEST”的Label﹐取名Lbest﹒在Lbest下面佈署1個顯示遊戲最高分的LCD Number﹐取名lcdbestScore﹒

以下為到目前為止的ui screen shot﹒



接著佈署Tile Map﹒因為要在每個Tile的4周顯示背景色﹐所以又放上1個Frame﹐取名Frame\_1\_1﹒依序佈署4列 X 4行 共16個Label在Frame\_1\_1上﹒建立1個gridLayout﹐並將16個Label配置給它﹒

程式中會將每個gridLayout中的Label依序設給tilesMap 2維陣列中對應Tile的private member QLabel label﹒每個Label的顏色和背景色﹐依其所屬Tile的另一private member int num內存數字而定﹐所以在程式中動態設定每個Tile的Label顏色和背景色

另外﹐設定gridLayout的layoutHorizontalSpacing為15﹐layoutVerticalSpacing為15﹒這樣水平方向和垂直方向﹐Label間都留有15點的空間﹐

到此程式的ui設計已經完成﹒

3.程式架構說明

3.1. main()函數建立遊戲視窗game2048物件w﹐w的constructor 執行了initgame()﹐initgame()執行大部分的程式初始設定﹐另一部分程式初始設定則在遊戲視窗物件w第一次顯示在螢幕後﹐在showEvent()函數裡做﹐因為這樣取得的tilesMap裡每個tile的Label x()﹐y()﹐width()﹐height()值比較正確﹒因為這個原因﹐本來應該由initgame()呼叫startGame()開始遊戲﹐也改在showEvent()裡執行第一次呼叫startGame()﹒從此以後﹐程式只會因玩家點了NewGame按鈕而執行startGame()﹒

3.2.因為要回應鍵盤按鍵事件﹐所以game2048物件添加了member function keyPressEvent()﹒每當玩家按了←﹑→﹑↑﹑↓鍵﹐就呼叫對應的遊戲控制函數﹐slideLeft()﹑slideRight()﹑slideUp()﹑slideDown()﹐實現遊戲功能﹒game2048類別的其它data member和member function大部分都是對應4個方向鍵的遊戲控制函數在其程序流程中依序會用到的﹒

3.3.因為要回應玩家點了NewGame按鈕﹐在設計程式的ui時﹐就將NewGame按鈕的signal函數clicked()和game2048的slot函數on\_PBNewGame\_clicked() connect

好了﹒所以每當玩家按了NewGame按鈕﹐就會呼叫on\_PBNewGame\_clicked()﹒我們在on\_PBNewGame\_clicked()裡面放了startGame()﹐所以就會重新開始遊戲﹒

3.4.這樣不管玩家點了NewGame按鈕﹐或是按了方向鍵﹐程式都能做出回應﹒點了NewGame按鈕就重新開始遊戲﹐按了方向鍵就由對應的遊戲控制函數做出應有的遊戲功能﹒



